Slider on a Timer

Putting Slides on a Timer

Converting a Slider

Taking one or more of the sliders you made earlier this week and changing it so that it runs on a timer, instead of clicking next and previous buttons is not too difficult.

Additional challenge: How can you make your slideshow stop changing slides, while a user hovers their mouse over the slides?

Remove Unneeded Bits

First thing to do is to remove the previous and next buttons from the HTML. You an also remove the the styles for these links from the stylesheet

```html
<div id="slider">
    <ul>
        <li><img src="slides/image1.jpg" alt="slideshow image"></li>
        <li><img src="slides/image2.jpg" alt="slideshow image"></li>
        <li><img src="slides/image3.jpg" alt="slideshow image"></li>
        <li><img src="slides/image4.jpg" alt="slideshow image"></li>
        <li><img src="slides/image5.jpg" alt="slideshow image"></li>
    </ul>
</div>

<p id="links"><a href="#" id="previous">previous</a><a href="#" id="next">next</a></p>
```

Remove Click Handlers

You can remove the previous button click handler entirely.

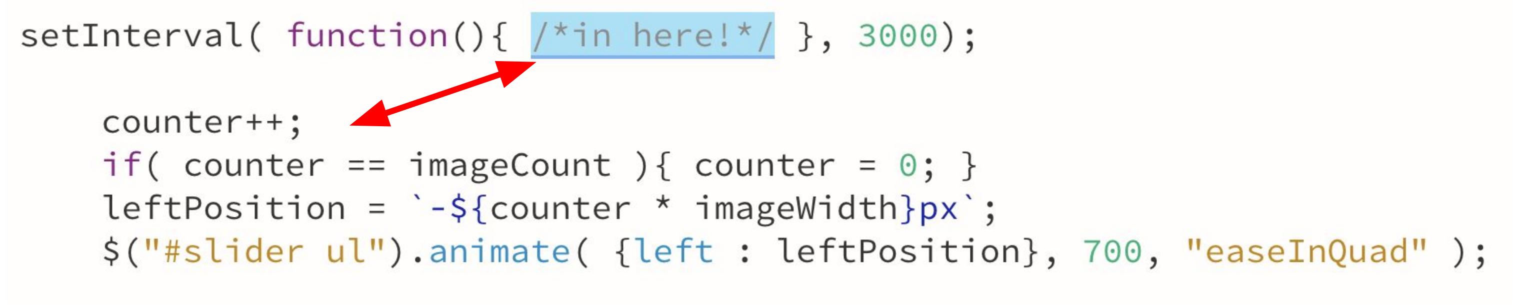
For the next button click handler, you just want to keep the guts of the click handler. Remove the first line and the closing braces.

```
$("#next").click( function(){

    counter++;

    if( counter == imageCount ){
        counter = 0;
    }

    leftPosition = `-${counter * imageWidth}px`;

    $("#slider ul").animate( {left : leftPosition}, 700, "easeInQuad" );

} );
```

Add the Setinterval Function

Add the setInterval function, which takes two parameters. The first is a function that you pass in, and the second is the amount of time before it goes on to the next slide.

You will put the guts of the next button click handler in the anonymous function that runs on the interval.



```
setInterval( function(){ /*in here!*/ }, 3000);

    counter++;
    if( counter == imageCount ){ counter = 0; }
    leftPosition = `-${counter * imageWidth}px`;
    $("#slider ul").animate( {left : leftPosition}, 700, "easeInQuad" );
```

Script with SetInterval

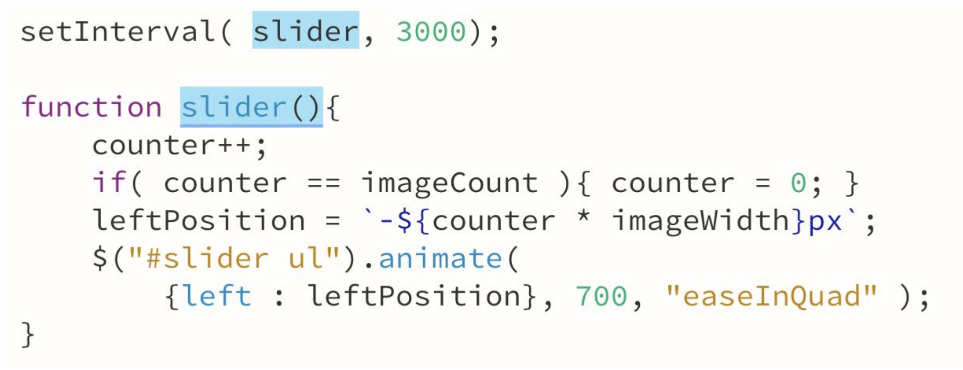
Here is the script, with the setInterval function finished.

```
$(window).on('load', function () {
      "use strict";

      const imageCount = $("#slider ul li").length;
      const imageWidth = $("#slider ul li:first img").width();
      const totalWidth = (imageCount * imageWidth) + "px";
      let leftPosition = 0;
      let counter = 0;

      $("#slider ul").css("width", totalWidth);

      setInterval(function(){
            counter++;
            if (counter == imageCount) { counter = 0; }
            leftPosition = `-${counter * imageWidth}px`;
            $("#slider ul").animate(
                  { left: leftPosition }, 700, "easeInQuad");
      }, 3000);

});
```

One More Feature

Sometimes sliders have a feature that when you roll the mouse over the slider, it pauses the timer.

To start this, pull the guts of the anonymous function back out and put them in a named function instead.

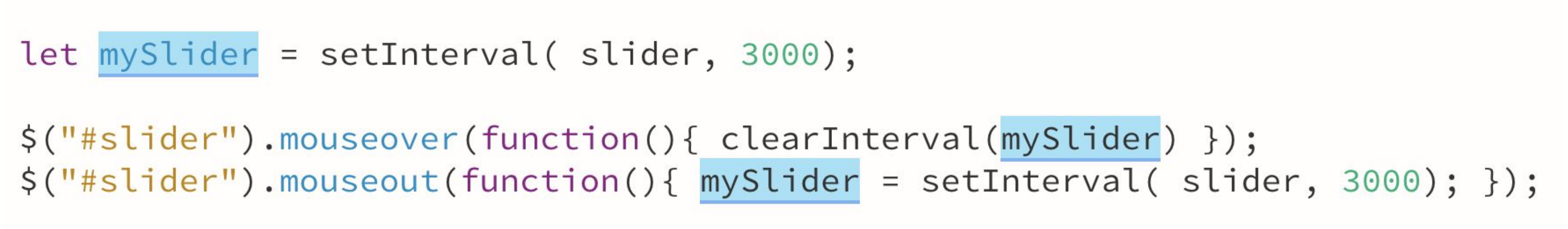


```
setInterval( slider, 3000);

function slider(){
    counter++;
    if( counter == imageCount ){ counter = 0; }
    leftPosition = `-${counter * imageWidth}px`;
    $("#slider ul").animate(
        {left : leftPosition}, 700, "easeInQuad" );
}
```

Add Event Handlers

Assigned the setInterval function to a variable called mySlider. That is then used in the mouseover event to stop the slider when the mouse is over the slider

When the mouse leaves the slider, it starts again (after waiting three seconds).



```
let mySlider = setInterval( slider, 3000);

$("#slider").mouseover(function(){ clearInterval(mySlider) });
$("#slider").mouseout(function(){ mySlider = setInterval( slider, 3000); });
```